the first channel command word. During the execution of the specified chain of I/O operations, data and further commands are transferred between the channel(s) and the control unit(s).

[0020] Fig. 2 is a schematic diagram of a single computer with shared physical memory 210, and may be an IBM z/Series z/900 computer available from International Business Machines Corporation of Armonk, New York which is a follow-on computer of the IBM S/390 computer. The computer is divided up into a number of logical partitions 212a -212n, each partition having discrete servers 214a-214n, respectively, labeled in Fig. 2 as discrete server 1 to discrete server n. Each discrete server has a TCP/IP layer 216a-216n, respectively, for handling the transmission protocols for transmitting data in Input/Output (I/O) operations for networks, as is well known. Under each TCP/IP layer 216a-216n is a device driver 218a-218n, respectively, for driving data transmissions between the discrete servers, as will be discussed.

[0021] In the present invention, each device driver is similar to device drivers which drive the devices 190 of Fig. 1. However the device drivers 218 of Fig. 2, rather than driving I/O devices, drive data exchanges between the LPAR partitions, as will be explained. Each device driver 218 has a send queue 222, and a receive or target queue 220; the send queue 222 being used for sending data from the respective discrete server 214 when that discrete server is the sending server, and the receive queue 220 for receiving data for its respective discrete server 214 when that discrete server is the target server in a send operation, as will be described in connection with Fig. 3. A common lookup table 224 is in the HSA portion 225 of the main storage 110 of the single computer 210 across the entire computer, as explained in Fig. 1. This common lookup table 224 is a centralized table defining the discrete servers 214a-214n within the computer 210 and is maintained in HSA 225 that is accessible by all the discrete servers 214a-214n. However, the discrete servers can only register in the common lookup table using I/O type commands, and cannot retrieve any information from the lookup table 224, thus maintaining security between the servers.

[0022] Each device driver 218 is associated with a subchannel control block 227 which contains control information for the subchannel. As is known, the subchannel control blocks exist in HSA

225 and are uniquely identified by a subchannel number. The subchannel control block 227 includes an internal queued direct I/O (IQDIO) indicator 228 which indicates if this subchannel is an IQDIO subchannel. The IQDIO indicator 228 may be set by the channel path identifier (CHPID) definition statement during the configuration process, as is well known in the art.

[0023] The architecture of the computer 210 of the present invention adheres to the queued direct I/O (QDIO) architecture, as explained in U.S. Patent Application Serial No. 09/253,246 filed February 19, 1999 by Baskey et al. for A METHOD OF PROVIDING DIRECT DATA PROCESSING ACCESS USING A QUEUED DIRECT INPUT-OUTPUT DEVICE, owned by the assignee of the present invention and incorporated herein by reference.

[0024] Fig. 3 is an illustration of the common lookup table 224 of Fig. 2, and includes hash tables control area 300, a source queue hash table 310, and a target queue hash table 320. The source queue hash table includes multiple entries starting with the first entry 311, each entry acting as a source queue duplicate list head (containing a pointer to duplicate list entries 312). The target hash table 320 includes multiple entries starting with the first entry 321, each entry acting as a target queue duplicate list head (containing a pointer to duplicate list entries 322). A common queue control area 330 is shared by both send (using table 310) and receive (using table 320) processing. It will be noted that multiple 322s can point to a single 330. Each queue control 330 is linked to a QDIO queue set 340. New entries in the source queue hash table 310 are created at 312, and new entries in the target queue hash table 320 are created at 322, as will be explained.

[0025] Fig. 4 is a block diagram of the hash table control 300 and includes a hash table shared serialization lock 401, and a hash table exclusive update lock 402. Fig. 5 is a block diagram of the queue control 330 and includes a QDIO pointer 430 which points to the queue set 340, an outbound lock 431, and an inbound lock 432.

[0026] Fig. 6 is a block diagram of the queue set 340 of Fig. 3 and includes a send queue 440 having multiple entries, and a receive queue 445 having multiple entries. The queue set 340 also

includes a storage list status block (SLSB) 442 which shows the status of each entry in the send queue 440, and a storage list status block (SLSB) 447 which shows the status of each entry in the receive queue 445. Each active entry of the send queue 440 has an associated buffer pointer 441 which points to a user buffer 443 for containing the data to be sent to the target LPAR partition. Fig. 7 is an illustration of the transfer data in the user buffer 243, and includes the target IP address 244 to which the data is to be sent. Each active entry in the receive queue 445 is associated with a buffer pointer 446 which points to a user buffer 448 which is to receive the data transferred from the user buffer 443.

[0027] Fig. 8 is a block diagram illustrating the entries of the source queue hash table list 310 as set up at 312. Each entry includes the LPAR-ID.SUBCHANNEL# 410 used as a key to the table 311, the status 411 of the entry, the queue control pointer 412 which points to the control 330 for this entry, a next pointer 413 which points to the next entry 312 in the source hash table 310, and a previous pointer 414 which points to either the first entry 311 in the source hash table 310 or the previous entry created at 312. Similarly, Fig. 9 is a block diagram illustrating the entries of the target queue hash table as set up at 322. Each entry includes the IP address 420 used as a key to the table 321, the status 421 of the entry, a queue control pointer 422 which points to the control 330 for this entry, a next pointer 423 which points to the next entry 322 in the target hash table 320, and a previous pointer 424 which points to either the first entry 321 in the target hash table 320 or the previous entry created at 322.

[0028] The first step in transferring data from one LPAR partition to another, is to register a source or send queue 222 (represented in Fig. 2 as a downward arrow, and also shown as queue 440 in Fig. 6) and a receive or target queue 220 (represented in Fig. 2 as an upward arrow, and also shown as queue 445 in Fig. 6) for a send transaction. The registration process includes three steps: the first is to register the QDIO queue set 340 (one send queue 222 and one target queue 220) in the source queue hash table 310; the second is to associate one or more IP addresses with the previously defined QDIO set 340 by adding entries to the target queue hash table 320; and the third is to define the I/O completion vector polling bytes (620a, 615a, and 612 to be discussed in connection with Fig. 10) that are to be used to pass initiative to the target. As each QDIO queue